# TO DO

(In no particular order at this time)

Make a list of components and state and figure out priority.

Build a card component with a super super basic frame, a spot to render current power dynamically, and the name, cost and description on hover/right click etc.

* A player can drag a card from their hand, into one of their not full CardZones, and it moves to the lowest number card position in that card zone. This updates (or maybe queues up the update depending on how we implement) the state of the card in said cardZone
* Single source of truth, object that holds all the game state logic, which is passed down as props through app components using a hook.
  + We can update state globally as soon as a change is made, but only render the changes after both players have locked in?
* implement a hook for card component similar to useVisualMode from scheduler.
  + Several cards affect either the last card played or next card you will play, so will need to have a history & that turns actions as a state.
* -able to draw cards from deck
* -able to play cards from hand
* Add ‘seeded’ game end point (/game/:id) where if both players input the same seed, they are both directed to the same game, connects via websockets and assigns each a player #
* Game renders differently for p1 or p2, but cards in game logic are still being ‘held’ in a single view.
* Cards placed in lanes are face up but turned face down when “end turn” selected (not visible to the other player); when both players have ended their turn, then they are turned upright and calculations of Pow and special abilities are made
* At the end of the sixth round final calculations are made and winner is determined as player who had more power in 2 of 3 of the lanes
* If there is lane with equal pow on both sides, and each player wins 1 other lane, it is a tie, no winners declared.

# 

# 

# ROUTES

* Home
  + Enter username, choose avatar, select your deck-halves and start a game
* In game - 2 modes waiting/in game.
* Post game -3 modes win/lose/draw

Snap up.

# THINGS TO ASK ANDY?

* Any framework suggestions for websockets?
* ORM suggestion- sequelize?
* Is our name too dumb?
* State management.

# 

———————————————————————————

General comments/concerns

* I think we should not bother with any ‘snap’ functionality, as we aren’t going to have matchmaking, or leaderboards/ranks, so is a bit redundant
* For decks we can go two routes, we can make a bunch of pre-made decks which I think should be our goal for now, and for stretch, try and make a deck creator/editor.
* <https://marvelsnap.io/> has a card database and location database we can scrape for our designs.
* Some cards will be much more complex to build than others, so we should prioritize which we build functionality for reusability first, and then more complex as we assess our time needs. Same goes for locations
* Cards and locations with Move may be very hard to implement, should probably be a stretch goal if any.
* We should start with 2 decks of 12 cards per deck (try to have them balanced with each other). Then add additional cards/decks as stretch goals. We could even start with 1 deck of 12 cards and have each player play the same deck to start to keep it simpler.
* I imagine our app as a ‘Goldfish simulator’ kind of tool, built for ‘testing decks’, so i’m unsure if we need any timer/scoreboard features other than a re-match function? Can totally scrap this concept.

### Technology

* I actually think phaser is the wrong call here. I did the intro tutorial, and messed around with the whole scenes thing, I actually think it will overkill and make it harder to develop functionality. I started to see if I could boiler plate a layout, feel free to take a look, it’s just vanilla create-react-app. (we don’t have to use any of this code, just wanted to see what I could do) <https://github.com/JadeDuo/react-card-game>
* <https://blog.logrocket.com/top-websocket-libraries-nodejs-2022/>
* ^ Need to decide on our server side framework
* <https://sailsjs.com/> We actually may want to consider sails, it has built in WS integration, uses node.js, we can use all the shortcuts for routing, and english adjacent sql calls.
* <https://feathersjs.com/> <-built ontop of node.js and express and socket.io, this would be a much lighterweight option than sails,

### Card Design State thoughts

* Several cards affect either the last card played or next card you will play, so will need to have a history like in interview scheduler
* Ongoing state can hold the abilities of cards with ongoing abilities as long as they are on the field
* On Reveal can trigger on change of state from the empty card slot to card identity

### Gameplay thoughts

* It may be easier to code each turn 0-6 (7 turns) separately, turn 0 for setup and drawing cards, since some cards trigger on specific turns, and turn 1-3 have location reveals.
* Having a controller for each lane is likely necessary, so we can keep track of lane based state
* In actual marvel snap, first reveal priority goes to the player who currently is winning the most lanes, and then the order you play cards is the order they reveal in. I’m unsure of how difficult/easy this will be to implement, but could probably as a player is playing cards for that turn, add that action into an array and resolve fifo? (Jesse edit: or as a placeholder logic, just swap turns like the multiplayer math game. Player 1 gets first reveal on turn one. Player 2 gets reveal priority on turn 2, etc. For 6 turn games everyone gets 3 priority rounds? Stretch could be by lane winning, etc)
* Timer on turns